REMARKS

Claims 1-30 are pending in the application. As required by 37 CFR § 1.121, Applicant submits a version with markings showing changes to the application. In light of the amendments and following remarks, Applicant believes all the pending claims are now in condition for allowance.

Formal Matters

The Office Action objected to Figures 1 and 2 alleging that a "Prior Art" legend is required. Applicant requests to amend these figures to include the legend as shown in red on the attached drawing sheet. Accordingly, the objection is overcome.

The Office Action rejected claims 6, 8, 15, 16, 21, 22, 27, and 28 under 35 USC § 112, second paragraph, as allegedly being indefinite. More specifically, the Office Action indicated that claims 6, 15, 21, and 27 lacked antecedent basis for "the thread." Applicant has amended these claims to recite "the first thread," which has proper antecedent basis, so the rejection is overcome.

The 35 USC § 103(a) Rejection of Claims 1-6 and 8-30

Claims 1-6 and 8-30 were rejected claims under 35 USC § 103(a) as being unpatentable over U.S. Patent No. 6,009,269, issued December 28, 1999 to Burrows et al. (hereinafter "Burrows"). For the following reasons, Applicants respectfully traverse the rejection as the reference does not support a prima facie case of obviousness to the current claims.

In a sincere effort to expedite prosecution, Applicant amended the claims to more clearly describe the invention. However, Applicant reserves all right to pursue the original or other claims in a continuing application.

The claims have been amended to recite, for example, that the first thread is suspended after it requests access to a resource that is available. This is contrary to conventional techniques, including the techniques described in Burrows, as threads are granted access to the resource if it is available (see, e.g., col. 3, lines 43-46 of Burrows). With the invention as recited in claim 1, a second thread can request access to the resource thereby potentially causing a race condition. Burrows does not disclose or suggest these features so the reference does not support a prima facie case of obviousness.

As all of the independent claims have been amended to include similar features, all the pending claims are allowable over the cited art Burrows.

The 35 USC § 103(a) Rejection of Claim 7

Claim 7 was rejected claims under 35 USC § 103(a) as being unpatentable over Burrows further in view of U.S. Patent No. 5,630,128, issued May 13, 1997 to Farrell et al. (hereinafter "Farrell"). Claim 7 is a dependent claim that incorporates all the features of the independent claim. As it has not been shown that the secondary reference Farrell remedies the deficiencies of the primary reference Burrows, the claim is allowable for the same reasons as above.

Conclusion

For the foregoing reasons, Applicant believes all the pending claims are in condition for allowance and should be passed to issue. If the Examiner feels that a telephone conference would in any way expedite the prosecution of the application, please do not hesitate to call the undersigned at (408) 446-8693.

Respectfully submitted,

Peggy A. Su (Reg. No. 41,336)
for Michael J. Ritter

Michael J. Ritter
Reg. No. 36,653

RITTER, LANG & KAPLAN LLP
12930 Saratoga Ave., Suite D1
Saratoga, CA 95070
Tel: 408-446-8690
Fax: 408-446-8691

VERSION WITH MARKINGS TO SHOW CHANGES MADE TO THE APPLICATION

In the Claims

Claims 1, 6, 9, 11, 15, 17, 19, 21, 23, 25, 27, and 29 have been amended as follows:

1. (Amended) A method of analyzing multi-threaded programs, comprising:

determining that unsynchronized accesses to a resource of interest can be performed by a plurality of threads;

receiving a request from a first thread to access the resource, wherein the resource is available;

suspending the first thread; and

while the first thread is suspended, receiving a request from a second thread to access the resource.

6. (Amended) The method of claim 5, wherein the first thread is also suspended on an event, meaning that the event awakens the first thread.

9. (Amended) A computer program product for analyzing multi-threaded programs, comprising:

computer code that determines that unsynchronized accesses to a resource of interest can be performed by a plurality of threads;

computer code that receives a request from a first thread to access the resource, wherein the resource is available;

computer code that suspends the first thread;

computer code that while the first thread is suspended, receives a request from a second thread to access the resource; and

a computer readable medium that stores the computer codes.

11. (Amended) A method of analyzing multi-threaded programs, comprising:

determining that unsynchronized accesses to a memory location can be performed by a plurality of threads;

receiving a request from a first thread to write data to the memory location, wherein the memory location is available for writing;

suspending the first thread; and

while the first thread is suspended, receiving a request from a second thread to write data to the memory location.

15. (Amended) The method of claim 14, wherein the first thread is also suspended on an event, meaning that the event awakens the first thread.

17. (Amended) A computer program product for analyzing multi-threaded programs, comprising:

computer code that determines that unsynchronized accesses to a memory location can be performed by a plurality of threads;

computer code that receives a request from a first thread to write data to the memory location, wherein the memory location is available for writing;

computer code that suspends the first thread;

computer code that while the first thread is suspended, receives a request from a second thread to write data to the memory location; and

a computer readable medium that stores the computer codes.

19. (Amended) A method of analyzing multi-threaded programs, comprising:

determining that unsynchronized accesses to a memory location can be performed by a plurality of threads;

receiving a request from a first thread to write data to the memory location, wherein the memory location is available for writing;

suspending the first thread;

while the first thread is suspended, receiving a request from a second thread to write data to the memory location;

awakening the first thread; and

logging for a user that the first and second thread performed unsynchronized writes to the memory location.

21. (Amended) The method of claim 20, wherein the first thread is also suspended on an event, meaning that the event awakens the first thread.

23. (Amended) A computer program product for analyzing multi-threaded programs, comprising:

computer code that determines that unsynchronized accesses to a memory location can be performed by a plurality of threads;

computer code that receives a request from a first thread to write data to the memory location, wherein the memory location is available for writing;

computer code that suspends the first thread;

computer code that while the first thread is suspended, receives a request from a second thread to write data to the memory location;

computer code that awakens the first thread;

computer code that logs for a user that the first and second thread performed unsynchronized writes to the memory location; and

a computer readable medium that stores the computer codes.

25. (Amended) A method of analyzing multi-threaded programs, comprising:

modifying an existing multi-threaded program include computer code that determines that unsynchronized accesses to a memory location can be performed by a plurality of threads;

modifying the existing multi-threaded program to include computer code that suspends a first thread that writes data to a memory location that is available, wherein a second thread writes data to the memory location; and

modifying the existing multi-threaded program to include computer code that logs for a user that the first and second thread performed unsynchronized writes to the memory location when a second thread writes data to the memory location.

27. (Amended) The method of claim 26, wherein the first thread is also suspended on an event, meaning that the event awakens the first thread.

29. (Amended) A computer program product for analyzing multi-threaded programs, comprising:

computer code that modifies an existing multi-threaded program include computer code that determines that unsynchronized accesses to a memory location can be performed by a plurality of threads;

computer code that modifies the existing multi-threaded program to include computer code that suspends a first thread that writes data to a memory location that is available, wherein a second thread writes data to the memory location;

computer code that modifies the existing multi-threaded program to include computer code that logs for a user that the first and second thread performed unsynchronized writes to the memory location when a second thread writes data to the memory location; and

a computer readable medium that stores the computer codes.